Soft-Output Trellis Waveform Coding

Tariq Haddad* and Abbas Yongaçoḡlu
School of Information Technology and Engineering, University of Ottawa
Ottawa, Ontario, K1N 6N5, Canada
Fax: +1 (613) 562 5175
thaddad@site.uottawa.ca *yongacog@site.uottawa.ca*

Abstract

In this paper, we exploit the similarity between source compression and channel decoding to provide a new encoding algorithm for trellis vector quantization (TVQ). We start by drawing the analogy between TVQ and the process of sequence-ML channel decoding. Then, the new search algorithm is derived based on the symbol-MAP decoding algorithm, which is used in soft-output channel decoding applications. Given a block of source output vectors, the new algorithm delivers a set of probabilities that describe the reliability of the different symbols at the encoder output for each time instant, in the minimum distortion sense. The performance of both the new algorithm and the Viterbi algorithm is compared using memoryless Gaussian and Gauss-Markov sources. The two algorithms provide expected similar distortion-rate results. This behavior is due to the fact that sequence-ML decoding is equivalent to symbol-MAP decoding of independent and identically distributed data symbols. In other words, delivering the sequence of minimum distortion symbols is equivalent to delivering the minimum distortion sequence of symbols.

1 Introduction

The efficiency of data transmission is directly related to the source compression process. Very often, we are interested in delivering information at rates lower than the entropy of the original source. This approach results in a loss in the fidelity of reproduction (distortion). The relation between the compression rate and the average distortion of reproduction is considered in the rate distortion theory. One of the main outcomes of this theory is that the performance of a source coding system approaches the distortion-rate theoretical limit as we increase the processing block length, even for memoryless sources [1]. This fact motivated the research in vector-based compression systems.

It has been proved that trellis waveform coding provide near theoretical limit distortion-rate performance [2], [3], [4]. A trellis vector quantization (TVQ) system is characterized by a finite-state machine (FSM) decoder and a trellis search encoding algorithm. The channel output sequence is fed into the FSM decoder which produces the corresponding index of a reproduction codeword. A copy of the FSM is used to construct the associated encoding trellis, whose branches are colored by the reproduction codewords. Then, we use a search algorithm to deliver the proper channel input sequence, which represents the branch indices of the minimum distortion path in the trellis.

There are several similarities between the areas of source coding and channel coding, which are mainly attributed to the duality between the rate distortion theory and the channel capacity dispute. Both the source encoder and the channel decoder are redundancy removal systems. Furthermore, both areas of research use the Euclidean space and employ lattice and trellis structures in the coding process. One of the algorithms that are used with TVQ is the Viterbi algorithm. The function of the Viterbi algorithm in searching for the minimum distortion path is analogous to its function in delivering the most probable transmitted sequence in channel decoding. In the context of channel decoding, it is possible to use either symbol-MAP or sequence-MAP algorithms, according to the application. Symbol-MAP decoding minimizes the probability of symbols in error, while sequence-MAP decoding minimizes the probability of sequences of symbols (words) in error. Nonetheless, for the special case of independent and identically distributed channel input symbols, the two algorithms are equivalent in the sense that both algorithms minimize the probability of sequence of symbols in error.

When we apply the Viterbi algorithm with TVQ (VTVQ), we do not have *a priori* knowledge about the statistics of the source encoder output. Therefore, we assume that the branch indices are independent and identically distributed in anticipation of producing an uncorrelated and uniformly distributed sequence of symbols; i. e., a redundancy free sequence. According to this assumption, and given a source output sequence of vectors, using an algorithm that produces a sequence of minimum distortion symbols is equivalent to using the Viterbi algorithm, which pro-

*The author is a full-time Ph. D. student.

duces the minimum distortion sequence.

In this paper, we introduce a new TVQ search algorithm. The new compression approach represents the "natural dual" to symbol-MAP trellis channel decoding. The search algorithm is based on the early BCJR algorithm introduced for channel decoding in [5]. The algorithm performs decision-delayed processing and delivers for each time instant a set of reliability (soft) values that are used to decide on the minimum distortion source encoder output symbol. Since the algorithm delivers soft information, in the form of transition probabilities, we call this encoding approach "soft trellis vector quantization" (STVQ). The equivalence between sequence-ML Viterbi and symbol-MAP encoding is shown in several simulation cases using memoryless Gaussian and Gauss-Markov sources. Although, the symbol-MAP algorithm is more complex than the Viterbi algorithm, it has the potential to be extended to match the source statistics. In order to benefit from the increased capacity of channels with memory, the symbol-MAP algorithm has been modified to fit the problem of decoding correlated channels [6]. Similarly, we can modify the new compression algorithm to use the source memory in order to provide improved distortion-rate results.

2 Trellis Vector Quantization

Without loss of generality, we will only consider deterministic FSM decoders [7]. The decoder is a lookup table addressed by a shift-register of constraint length ν and k-bit inputs. Accordingly, the TVQ is characterized by a trellis that has $M = 2^{k(\nu-1)}$ states, and 2^k branches stemming out of each state. The branches are colored or labeled by the indices that address a reproduction codebook $\mathcal{C} = \{c_0, c_2, ..., c_{K-1}\}$, where $K = 2^{k\nu}$. Encoding is performed using a trellis search algorithm, by first deciding on the minimum distortion path that corresponds to a sequence of source output vectors. Then, the encoder generates the branch indices $\{x_0, x_1, \ldots\}$ that constitute the minimum distortion path. Given that a trellis quantizer is encoding L-dimensional source output vectors, then the compression rate is simply $R = k/L$ bit/sample.

3 Source Encoding and Channel Decoding

Consider a channel decoder that operates on the channel output sequence Y_1^N to produce the maximum *a posteriori* probability sequence $\hat{X}_1^N$. This sequence represents the best estimate for the sequence X_1^N at the channel encoder input in the Bayesian sense. In the process of estimating $\hat{X}_1^N$, the channel decoder removes the controlled redundancy, which was introduced for error protection. Following a similar approach, we use a source encoder to remove the redundancy in the information source and deliver the sequence of symbols X_1^N. Therefore, we can imagine the information source as some sort of a communications channel, which corrupts a sequence of symbols with some sort of noise. TVQ encoding is performed by choosing the minimum distortion sequence according to

$$X_1^N = \arg \min_{X_1^N \in \mathcal{A}^N} D\{X_1^N | Y_1^N\} \tag{1}$$

where

$$D\{X_1^N | Y_1^N\} = \sum_{n=1}^{N} d(Y_n, x_n | s_{n-1}) \tag{2}$$

and $d(Y_n, x_n|s_{n-1})$ is the Euclidian distance between the source output vector Y_n and the codeword labeling the branch with index $x_n \in \mathcal{A} = \{0, 1, \ldots, 2^k - 1\}$, given the state s_{n-1}. Accordingly, the Viterbi algorithm searches for the minimum distortion path by choosing, at each time instant n, the branch labeled by x_{n-T}, which produces the minimum distortion, where T represents the encoding depth. The search is proceeded between a state s_{n-1} and another state s_n using the following metric

$$M_n = \min_{x_n \in \mathcal{A}} \{\ M_{n-1} + d(Y_n, x_n|s_{n-1})\ \} \tag{3}$$

where M_{n-1} is the metric (distortion) at state s_{n-1}.

On the other hand, MAP-decoding is performed using the following rule

$$\hat{X}_1^N = \arg \max_{X_1^N \in \mathcal{A}^N} \Pr\{X_1^N | Y_1^N\} \tag{4}$$

For the case of independent and identically distributed channel encoder input symbols, Eq. (4) reduces to

$$\hat{X}_1^N = \arg \max_{x_n \in \mathcal{A}} \prod_{n=1}^{N} \Pr\{Y_n | x_n, s_{n-1}\} \tag{5}$$

Since the logarithm is a monotonic function and the probability values are always positive, Eq. (5) is equivalent to

$$\hat{X}_1^N = \arg \max_{x_n \in \mathcal{A}} \sum_{n=1}^{N} \log p(Y_n | x_n, s_{n-1}) \tag{6}$$

where $p(Y_n|x_n, s_{n-1})$ is the transition probability between the channel output Y_n and the channel input, which is associated with channel encoder input x_n and encoder state s_{n-1}. It follows that, in order to implement the decision-rule of Eq. (6), we may use the Viterbi algorithm to follow the channel encoder trellis with the branch metric

$$M_n = \max_{x_n \in \mathcal{A}} \{\ M_{n-1} + \log p(Y_n|x_n, s_{n-1})\ \} \tag{7}$$

where M_{n-1} and M_n are the metrics at states s_{n-1} and s_n, respectively.

The similarity between the two cases of employing the Viterbi algorithm for source encoding and channel decoding is mathematically supported by Eqs. (3) and (7). In fact, if we assume that the channel probability distribution is

$$p(Y_n|x_n, s_{n-1}) = \exp\{-d(Y_n, x_n|s_{n-1})\} \quad (8)$$

and we substitute this value of $p(Y_n|x_n)$ in Eq. (7), we get

$$M_n = \max_{x_n \in \mathcal{A}} \{ \ M_{n-1} - d(Y_n, x_n|s_{n-1}) \ \} \quad (9)$$

Note that the metric in Eq. (9) is identical to the metric in Eq. (3). Hence, the problem of trellis vector quantization is equivalent to the decoding problem of a convolutionally encoded sequence transmitted over a channel characterized by Eq. (8).

The conditional pdf given by Eq. (8) represents an association probability that describes the degree of membership of the the vector Y_n to the set represented by the codeword addressed by x_n, s_{n-1}. Moreover, this distribution is a member of a family of distributions known as the Gibbs distribution, which represents the solution to an optimization problem in fuzzy clustering [8]. The Gibbs distribution is controlled by a Lagrange multiplier η as

$$p(Y_n|x_n, s_{n-1}) = \frac{\exp\{-\eta d(Y_n, x_n|s_{n-1})\}}{Z} \quad (10)$$

where Z is a normalizing factor. It is clear that this distribution assigns higher probability values to lower energy configurations.

4 Soft Trellis Vector Quantization

In this section we use the analogy developed thus far, between the source and the channel, to introduce a new TVQ search algorithm. The algorithm is based on the fact that we may consider the TVQ encoding process as convolutional decoding at the output of the channel defined by Eq. (8). Consider a stationary discrete-time source with output symbol Y_n at time n. Let the N-length output sequence be $Y_1^N = \{Y_1, Y_2, \ldots, Y_N\}$, where $Y_n = [y_{n1}, y_{n2}, \ldots, y_{nL}]$. Furthermore, the TVQ output is a branch index $x_n \in \mathcal{A} = \{0, 1, \ldots, I-1\}$, where $I = 2^k$. In order to produce a sequence of minimum distortion symbols, we first assume that the source is the channel described by Eq. (8). Then, given a source output sequence Y_1^N, we choose an output symbol according to

$$x_n = \text{arg} \max_{i \in [0, I-1]} \text{Pr} \ \{x_n = i|Y_1^N\} \quad (11)$$

The forward-backward procedure described in [5] is used to solve for the set of probabilities Pr $\{x_n = i|Y_1^N\}$, $i = 0, 1, \ldots, I-1$. We proceed by defining the joint probability variable

$$\lambda_n^i(m) = \text{Pr} \ \{x_n = i, s_n = m, Y_1^N\} \quad (12)$$

It follows that

$$\text{Pr} \ \{x_n = i|Y_1^N\} = \frac{\sum_{m=0}^{M-1} \lambda_n^i(m)}{\sum_{m=0}^{M-1} \sum_{i=0}^{I-1} \lambda_n^i(m)} \quad (13)$$

Recall that the problem of TVQ with the Viterbi algorithm is similar to Viterbi decoding at the output of a memoryless channel. Therefore, based on the same analogy the source is considered memoryless in the derivation of the new algorithm. Accordingly, we write Eq. (12) in the form

$$\lambda_n^i(m) = \alpha_n^i(m) \cdot \beta_n(m) \quad (14)$$

where

$$\left. \begin{array}{rcl} \alpha_n^i(m) & = & \text{Pr} \ \{x_n = i, s_n = m, Y_1^n\} \\ \beta_n(m) & = & \text{Pr} \ \{Y_{n+1}^N|s_n = m\} \end{array} \right\} \quad (15)$$

It can be shown that the forward variable is evaluated using

$$\alpha_n^i(m) = \gamma_n^i(S_b^i(m)) \sum_{j=0}^{I-1} \alpha_{n-1}^j(S_b^i(m)) \quad (16)$$

where $S_b^i(m)$ is the state at which we arrive if we go backwards from state $s_n = m$ along the branch $x_n = i$. Besides,

$$\gamma_n^i(m) = \text{Pr} \ \{Y_n|x_n = i, s_{n-1} = m\} \quad (17)$$

We use Eq. (8) to substitute for the channel transition probability in Eq. (17), as follows

$$\gamma_n^i(m) = \exp\{-\eta d(Y_n, x_n = i|s_{n-1} = m)\} \quad (18)$$

The constant η is introduced in Eq. (18) in order to control the algorithm performance, as we will demonstrate in the simulation section.

The backward variable is computed as

$$\beta_{n-1}(m) = \sum_{j=0}^{I-1} \beta_n(S_f^j(m)) \cdot \gamma_n^j(m) \quad (19)$$

where $S_f^j(m)$ is the state at which we arrive if we go forward from state $S_{n-1} = m$ along the branch $x_n = j$.

5 Implementation Approach

One critical point to consider is the initialization of $\alpha_0^i(m)$ and $\beta_N(m)$. In our simulations we start the processing of each block of vectors from the zero state, thus, we initialize the forward variable as

$$\alpha_0^i(m) = \left\{ \begin{array}{cc} \frac{1}{2^k} & m = 0 \\ 0 & m \neq 0 \end{array} \right. \quad \forall \, i \quad (20)$$

Then, we proceed in the computation of $\alpha_n^i(m)$ for $n = 1, 2, \ldots, N$ according to Eq. (16). On the other hand, initializing $\beta_N(m)$ depends on whether we know the final state

Table 1: The VTVQ and STVQ 1-bit coding performance for the memoryless Gaussian source.

	STVQ		VTVQ	
ν	SNR (dB)	CI (dB)	SNR (dB)	CI (dB)
2	4.747	± 0.005	4.734	± 0.005
3	4.867	± 0.004	4.815	± 0.005
4	5.203	± 0.004	5.108	± 0.006
5	5.185	± 0.004	5.145	± 0.004
6	5.325	± 0.004	5.320	± 0.004

Table 2: The VTVQ and STVQ 1-bit coding performance for the AR(1) source, $\rho = 0.9$.

	STVQ		VTVQ	
ν	SNR (dB)	CI (dB)	SNR (dB)	CI (dB)
2	6.821	± 0.127	6.864	± 0.125
3	8.662	± 0.067	8.655	± 0.084
4	10.051	± 0.058	9.976	± 0.058
5	10.794	± 0.029	10.744	± 0.024

or not. Practically, we can choose to start encoding from a specific state, however, we cannot choose the last state, which is controlled by the source output sequence. Thus

$$\beta_N(m) = \frac{1}{M}, \quad \forall\, m \tag{21}$$

After that, we compute $\beta_n(m)$ for $n = N, N-1, \ldots, 1$ using Eq. (19). After evaluating $\alpha_n^i(m)$ and $\beta_n(m)$, we compute $\lambda_n^i(m)$ using Eq. (14), then, we evaluate the set of probabilities of interest using Eq. (13). Encoding is performed by delivering the $x_n = i$ associated with the maximum probability value, as indicated in Eq. (11).

It is clear that the values of the forward and backward variables depend geometrically on past and future values of large number of terms. Consequently, it is very likely that their values get significantly large or vanishingly small. To solve this problem we scale each $\alpha_n^i(m)$ and $\beta_n(m)$ by $\sum_i \sum_m \alpha_n^i(m)$ and $\sum_m \beta_n(m)$, respectively. The same approach is successfully adopted in [9]. This normalization step compensates for the fact that we are not using the variable Z, which is used in Eq. (10).

6 Simulation Results

We provide in this section the rate-distortion simulation results of encoding a memoryless Gaussian source, and a Gauss-Markov source. The performance results are measured in terms of signal to MSE quantization noise ratios (SNR). A training sequence of $30,000$ samples is used to search for a locally optimal codebook using the known LBG-algorithm [7]. In order to initialize the codebook search process, we use the techniques mentioned in [10]. For the memoryless Gaussian source, we use a random initial codebook drawn from a zero-mean and 0.75 variance Gaussian distribution. On the other hand, we initialize the trellis branches with $\pm\sigma_y$ for the Gauss-Markov source, where σ_y^2 is the source variance. Then, we employ the generated codebook to find an average SNR value as well as the corresponding 95 percent confidence interval, using a 100 sequences of length 1500 samples each. The computation of the confidence interval is based on the fact that the computed distortion for each sample sequence is a sample average. Thus, we can apply the central limit theorem to claim that the distribution of the computed sample distortions is Gaussian. Accordingly, the mean of the sample average distortions is used to compute the average SNR, and the variance is used to compute the confidence interval.

In the first case study we consider the $R = k/L = 1/1 = 1$ bit/sample encoding of a memoryless unit-variance Gaussian source. The MSE-distortion rate function of a Gaussian source is given by

$$D_G(R) = 2^{-2R}\sigma_y^2 \tag{22}$$

where R is the compression rate in bits/sample. Thus, for a unit-variance source and rate 1 bit/sample, $D_G(1) = 0.25$, or equivalently 6.02 dB. In Table 1, we show the simulation results for several trellis constraint lengths ν, where we observe the close performance of the two algorithms. In Tables 2, 3, we demonstrate the performance of the two approaches applied to encoding a first order Gauss-Markov source with correlation coefficient $\rho = 0.9$. In Table 2, we encode at a rate $R = k/L = 1/1$ bit/sample for several values of ν, while the results in Table 3 are for $R = 1/2$ bit/sample. The MSE-distortion rate function for a first order Gauss-Markov source is

$$D_G = 2^{-2R}(1-\rho^2)\sigma_y^2 \tag{23}$$

Accordingly, $D_G(0.5) = 10.22$ dB, and $D_G(1) = 13.23$ dB. In the following section we discuss the simulation results and comment on the algortihms' behavior.

7 Discussion

We have applied in this paper the forward-backward symbol-MAP algorithm in the area of data compression. The new algorithm delivers a sequence of minimum distortion symbols, which is equivalent to the minimum distortion sequence generated by the Viterbi algorithm. Experimental results showed that an encoding depth of at least $80k\nu$ vectors is required for the new algorithm to perform as well as the Viterbi algorithm. On the other hand, the survivor paths of the Viterbi algorithm seem to converge after processing $40k\nu$ vectors. The Lagrange multiplier η (refer to Eq. (18)) controls the degree of association of the source output vectors to the different encoding sets. It is shown in the literature of Fuzzy clustering that the value of η is related to a soft average distortion measure [8]. In the context of this paper,

Table 3: The VTVQ and STVQ 1/2-bit coding performance for the AR(1) source, $\rho = 0.9$.

	STVQ		VTVQ	
ν	SNR (dB)	CI (dB)	SNR (dB)	CI (dB)
2	5.981	± 0.082	6.049	± 0.121
3	7.408	± 0.058	7.475	± 0.043
4	8.280	± 0.037	8.224	± 0.026
5	8.685	± 0.011	8.569	± 0.019

we found that there is an optimal upper value of η for each case, after which the algorithm fails to converge to an "acceptable" optimal encoding configuration (codebook). This value was usually proportional to $\nu/(DL)$, where D is the expected MSE distortion of reproduction.

Although the STVQ algorithm is more complex than the Viterbi algorithm, a simplified version of the forward-backward algorithm [11] can ease the complexity issue. Nevertheless, the STVQ algorithm has the potential to be extended to use the source statistics (source memory) in the encoding process. Note that, we are not completely using the memory of the source in vector-based compression, since the vectors themselves are correlated. For example, we still have a potential of 1.5 dB of improvement using a constraint-length 5 trellis (Table 3), and the gap increases with lower constraint-length trellises (lower complexity). This motivates the extension of the symbol-MAP algorithm to make use of the available resources by projecting the source memory onto the trellis encoding path.

References

[1] T. Berger. *Rate Distortion Theory, A Mathematical Basis For Data Compression*. Englewood Cliffs, NJ: Prentice-Hall, 1971.

[2] F. Jelinek. "Tree encoding of memoryless time-discrete sources with a fidelity criterion". *IEEE Trans on Information Theory*, IT-15(5), Sep. 1969.

[3] A. J. Viterbi and J. K. Omura. "Trellis encoding of memoryless discrete-time sources with a fidelity criterion" . *IEEE Trans. on Information Theory*, IT-20:325–332, May 1974.

[4] R. M. Gray. "Time-invariant trellis encoding of ergodic discrete-time sources with a fidelity criterion" . *IEEE Trans. on Information Theory*, IT-23(1):71–83, Jan. 1977.

[5] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv. "Optimal decoding of linear codes for minimizing symbol error rate" . *IEEE Trans. on Information Theory*, IT-20(2):284–287, Mar. 1974.

[6] I. D. Marsland and P. T. Mathiopoulos. "Multiple differential detection of parallel concatenated convolutional (turbo) codes in correlated fast rayleigh fading " . *IEEE Journal On Selec. Areas In Commun.*, 16(2):265–274, Feb. 1998.

[7] L. C. Stewart, R. M. Gray, and Y. Linde. "The design of trellis waveform coders". *IEEE Trans. on Commun.*, COM-30(4):702–710, April 1982.

[8] K. Rose, E. Gurewitz, and G. C. Fox. "Vector quantization by deterministic annealing" . *IEEE Trans. on Information Theory*, 38(4):230–239, Jan. 1996.

[9] L. R. Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition" . *Proc. of the IEEE*, 77(2):257–286, Feb. 1989.

[10] E. Ayanoğlu and R. M. Gray. "The design of predictive trellis waveform coders using the generalized Lloyd algorithm". *IEEE Trans. on Commun.*, COM-34(11):1073–1080, Nov. 1986.

[11] A. J. Viterbi. "An intuitive justification and a simplified implementation of the MAP decoder for convolutional codes" . *IEEE Journal On Selec. Areas In Commun.*, 16(2):260–264, Feb. 1998.